PCT/GB2003/004533

10/530472

The Patent Office

INVESTOR IN PEOPLE

The Patent Office
Concept House
Cardiff Road
Newport
South Wales
NP10 8QQ

PRIORITY DOCUMENT

SUBMITTED OR TRANSMITTED IN COMPLIANCE WITH RULE 17.1(a) OR (b)

REC'D 24 NOV 2003	
WIPO	PCT

I, the undersigned, being an officer duly authorised in accordance with Section 74(1) and (4) of the Deregulation & Contracting Out Act 1994, to sign and issue certificates on behalf of the Comptroller-General, hereby certify that annexed hereto is a true copy of the documents as originally filed in connection with the patent application identified therein.

In accordance with the Patents (Companies Re-registration) Rules 1982, if a company named in this certificate and any accompanying documents has re-registered under the Companies Act 1980 with the same name as that with which it was registered immediately before re-registration save for the substitution as, or inclusion as, the last part of the name of the words "public limited company" or their equivalents in Welsh, references to the name of the company in this certificate and any accompanying documents shall be treated as references to the name with which it is so re-registered.

In accordance with the rules, the words "public limited company" may be replaced by p.l.c., plc, P.L.C. or PLC.

Re-registration under the Companies Act does not constitute a new legal entity but merely subjects the company to certain additional company law rules.

Signed

Dated 4 November 2003

BEST AVAILABLE COPY

Patents Form 1/77

Patents Act 1977
(Rule 16)

Request for grant of a patent

(See the notes on the back of this form. You can also get an explanatory leaflet from the Patent Office to help you fill in this form)

The Patent Office

THE PATENT OFFICE
D
29 OCT 2002
LONDON

1/77

30OCT02 E759411-1 D03052
P01/7700 0.00-0225139.5

The Patent Office
Cardiff Road
Newport
Gwent NP10 8QQ

1.	Your reference	A30216
2.	Patent application number *(The Patent Office will fill in this part)*	0225139.5 29 OCT 2002
3.	Full name, address and postcode of the or of each applicant *(underline all surnames)*	BRITISH TELECOMMUNICATIONS public limited company 81 NEWGATE STREET LONDON, EC1A 7AJ, England Registered in England: 1800000
	Patents ADP number *(if you know it)*	1867002
	If the applicant is a corporate body, give the country/state of its incorporation	UNITED KINGDOM
4.	Title of the invention	Method and Apparatus for Network Management
5.	Name of your agent *(if you have one)*	ROBINSON, Simon Benjamin
	"Address for Service" in the United Kingdom to which all correspondence should be sent *(including the postcode)*	BT GROUP LEGAL SERVICES INTELLECTUAL PROPERTY DEPARTMENT HOLBORN CENTRE 120 HOLBORN LONDON, EC1N 2TE
	Patents ADP number *(if you know it)*	~~1867001~~ 7980311001
6.	If you are declaring priority from one or more earlier patent applications, give the country and the date of filing of the or of each of these earlier applications and *(if you know it)* the or each application number	Country / Priority application number *(if you know it)* / Date of filing *(day / month / year)*
7.	If this application is divided or otherwise derived from an earlier UK application, give the number and the filing date of the earlier application	Number of earlier application / Date of filing *(day/month/year)*
8.	Is a statement of inventorship and of right to grant of a patent required in support of this request? *(Answer 'Yes' if:* *a) any applicant named in part 3 is not an inventor, or* *b) there is an inventor who is not named as an applicant, or* *c) any named applicant is a corporate body.* *(See note (d))*	YES

9. Enter the number of sheets for any of the following items you are filing with this form. Do not count copies of the same document

Continuation sheets of this form	
Description	12
Claim(s)	0
Abstract	1
Drawing(s)	7 + 7

8

10. If you are also filing any of the following, state how may against each item

Priority Documents	
Translations of priority documents	
Statement of inventorship and right to grant of a patent *(Patents Form 7/77)*	
Request for preliminary examination and search *(Patents Form 9/77)*	
Request for substantive examination *(Patents Form 10/77)*	
Any other documents *(please specify)*	

11. I/We request the grant of a patent on the basis of this application.

Signature(s): Date:

29 October 2002

ROBINSON, Simon Benjamin, Authorised Signatory

12. Name and daytime telephone number of person to contact in the United Kingdom **Samantha Radley** **020 7492 8146**

Warning

After an application for a patent has been filed, the Comptroller of the Patent Office will consider whether publication or communication of the invention should be prohibited or restricted under Section 22 of the Patents Act 1977. You will be informed if it is necessary to prohibit or restrict your invention in this way. Furthermore, if you live in the United Kingdom, Section 23 of the Patents Act 1977 stops you from applying for a patent abroad without first getting written permission from the Patent Office unless an application has been filed at least 6 weeks beforehand in the United Kingdom for a patent for the same invention and either no direction prohibiting publication or communication has been given, or any such direction has been revoked.

Notes

a) *If you need help to fill in this form or you have any questions, please contact the Patent Office on 0645 500505.*

b) *Write your answers in capital letters using black ink or you may type them.*

c) *If there is not enough space for all the relevant details on any part of this form, please continue on a separate sheet of paper and write "see continuation sheet" in the relevant part(s). Any continuation sheet should be attached to this form.*

d) *If you have answered 'Yes' Patents Form 7/77 will need to be filed.*

e) *Once you have filled in the form you must remember to sign and date it.*

f) *For details of the fee and ways to pay please contact the Patent Office.*

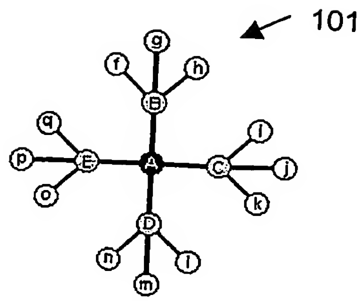


Figure 1a

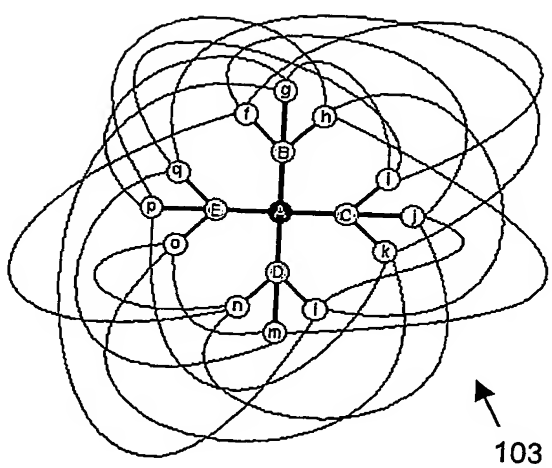


Figure 1b

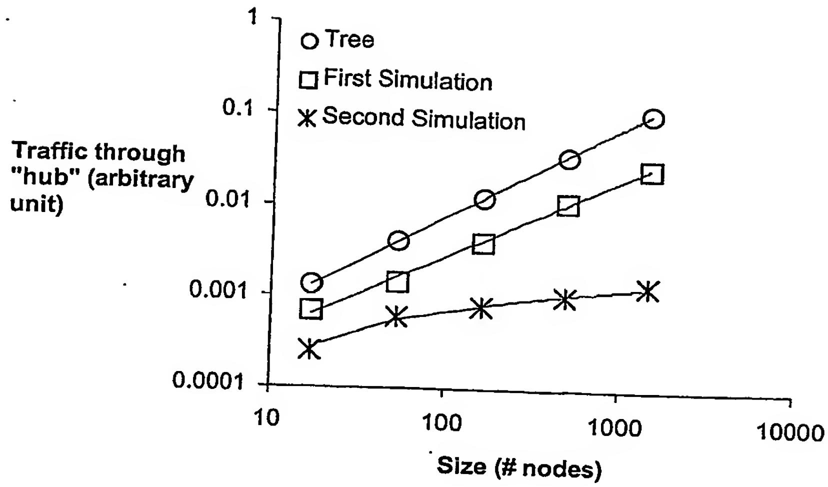

1
0.1
0.01
0.001
0.0001
Traffic through "hub" (arbitrary unit)
Tree
First Simulation
Second Simulation
10
100
1000
10000
Size (# nodes)

Figure 2

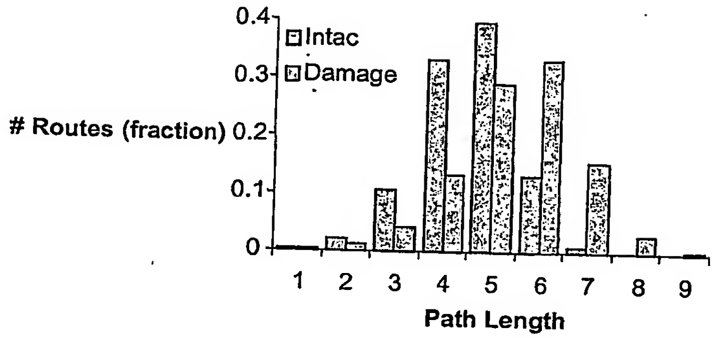


Figure 3a

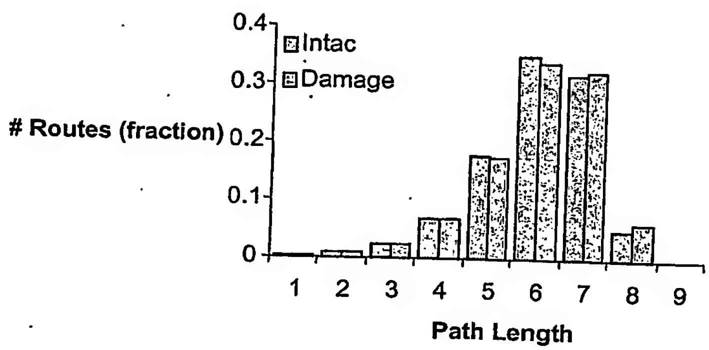


Figure 3b

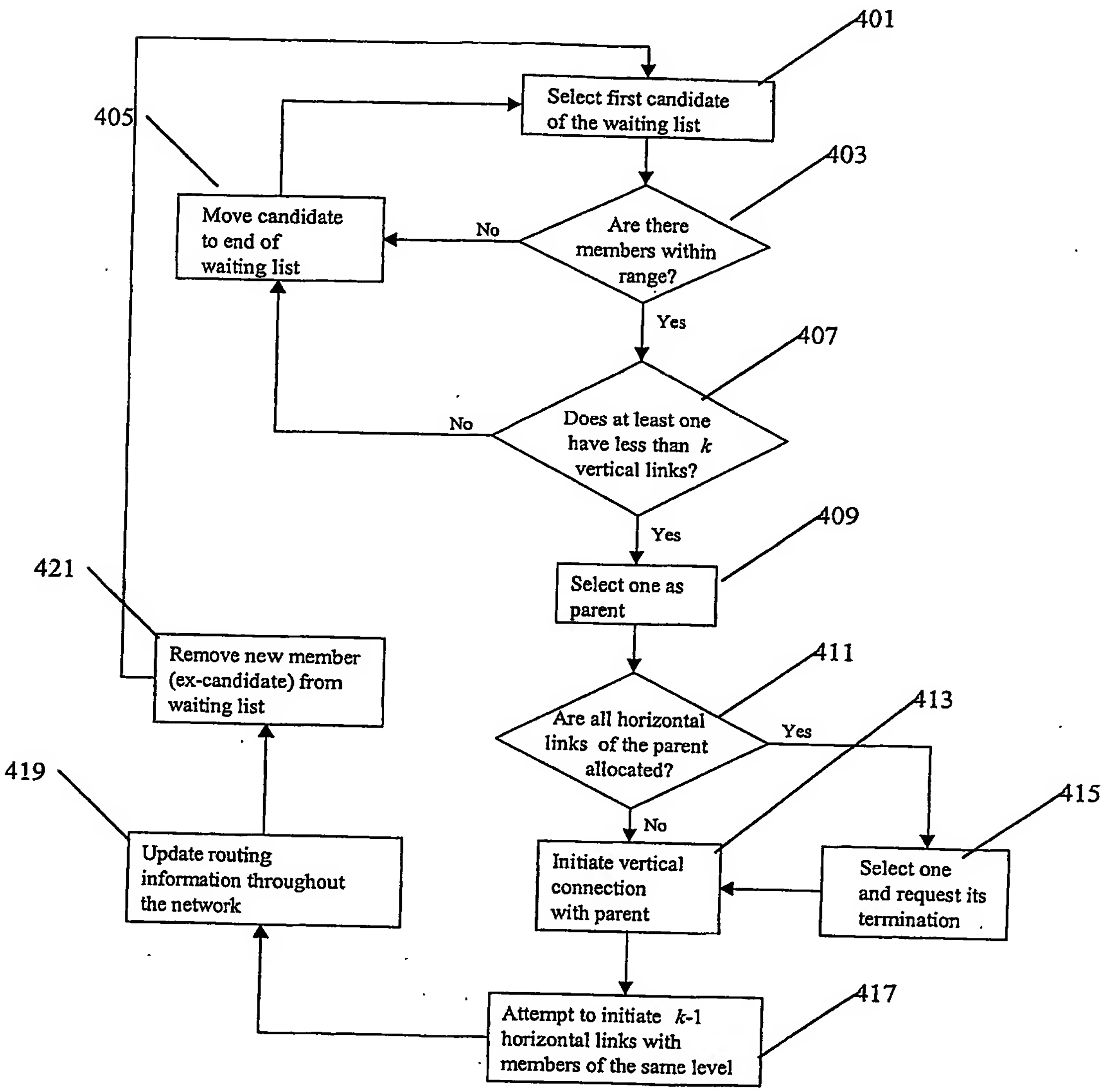

401
Select first candidate of the waiting list
403
Are there members within range?
No
405
Move candidate to end of waiting list
Yes
407
Does at least one have less than k vertical links?
No
Yes
409
Select one as parent
421
Remove new member (ex-candidate) from waiting list
411
Are all horizontal links of the parent allocated?
Yes
413
No
415
Select one and request its termination
Initiate vertical connection with parent
419
Update routing information throughout the network
417
Attempt to initiate k-1 horizontal links with members of the same level

Figure 4

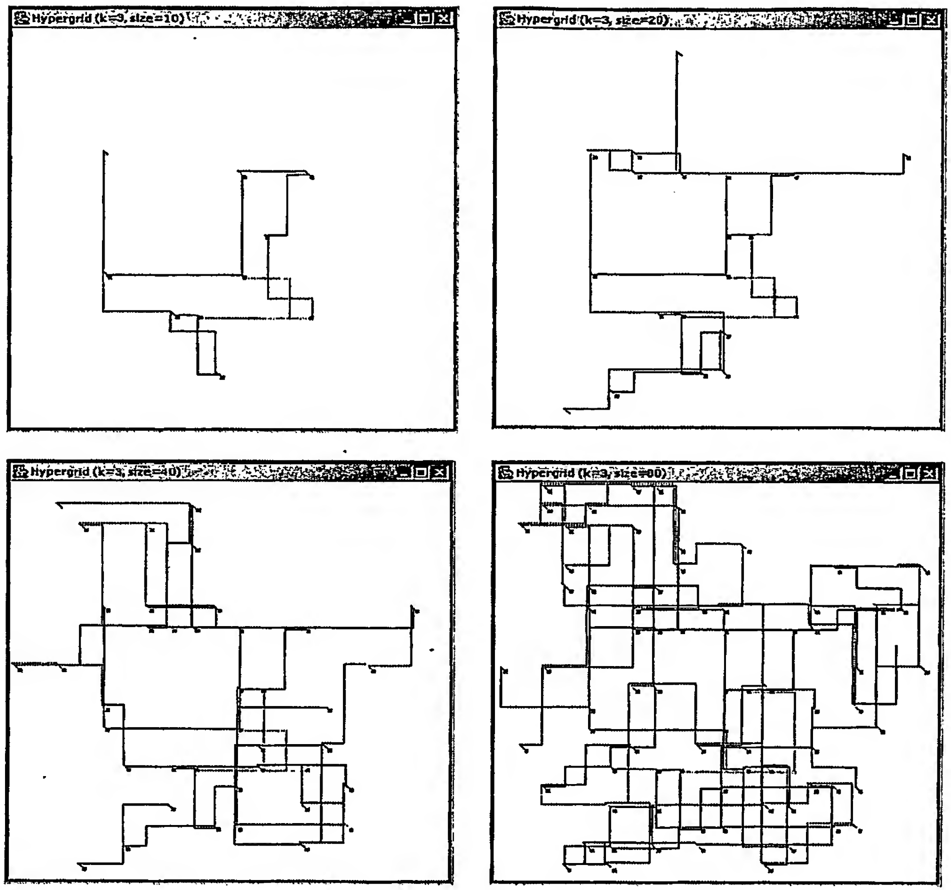

Hypergrid (k=3, size=10)
Hypergrid (k=3, size=20)
Hypergrid (k=3, size=40)
Hypergrid (k=3, size=80)

Figure 5

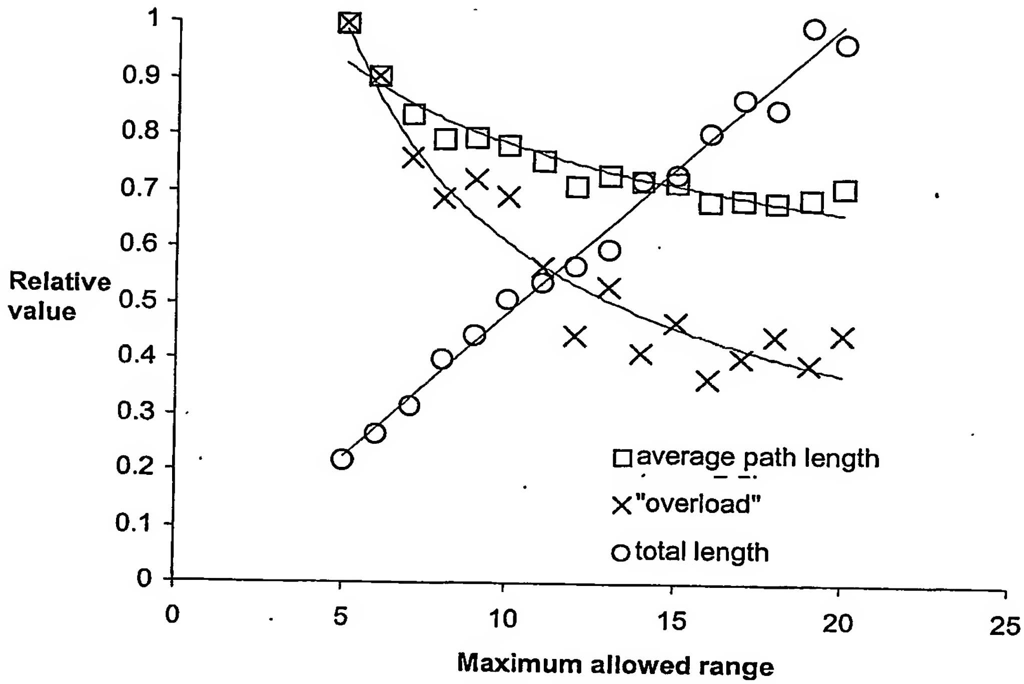

1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
Relative value
0
5
10
15
20
25
Maximum allowed range
average path length
"overload"
total length

Figure 6

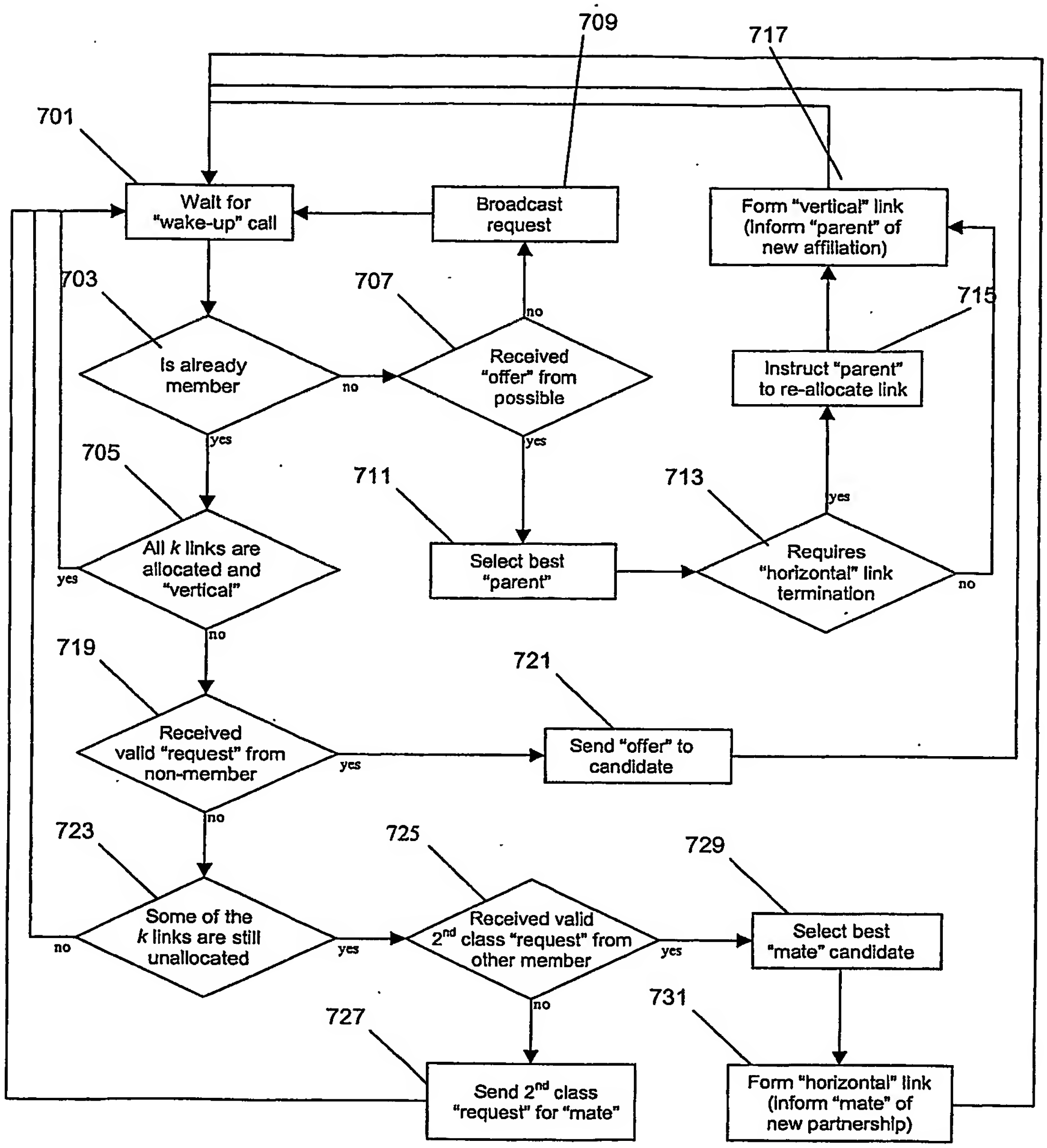

709
717
701
Wait for "wake-up" call
Broadcast request
Form "vertical" link (inform "parent" of new affiliation)
703
707
715
no
Is already member
no
Received "offer" from possible
Instruct "parent" to re-allocate link
yes
yes
705
711
713
yes
All k links are allocated and "vertical"
yes
Select best "parent"
Requires "horizontal" link termination
no
no
719
721
Received valid "request" from non-member
yes
Send "offer" to candidate
no
723
725
729
Some of the k links are still unallocated
no
yes
Received valid 2nd class "request" from other member
yes
Select best "mate" candidate
no
727
731
Send 2nd class "request" for "mate"
Form "horizontal" link (inform "mate" of new partnership)

Figure 7

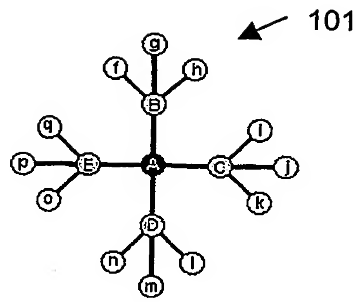


Figure 1a

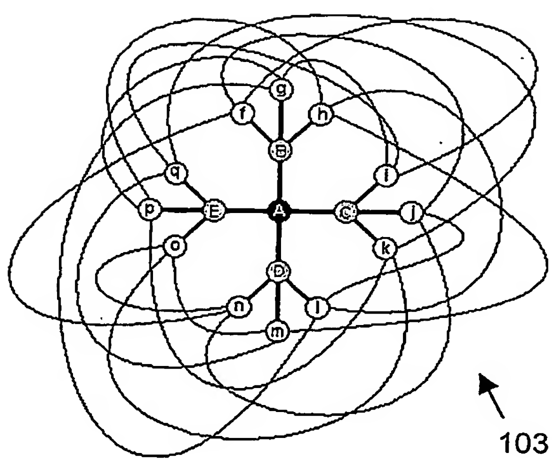


Figure 1b

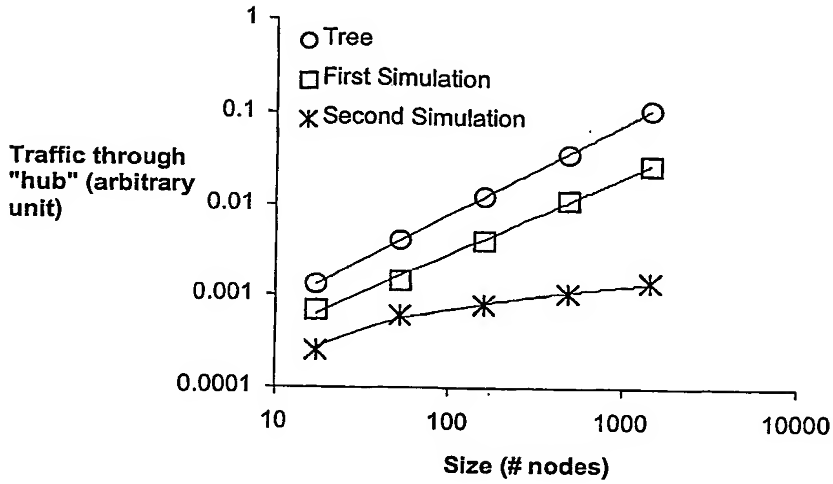

1
0.1
0.01
0.001
0.0001
Tree
First Simulation
Second Simulation
Traffic through "hub" (arbitrary unit)
10
100
1000
10000
Size (# nodes)

Figure 2

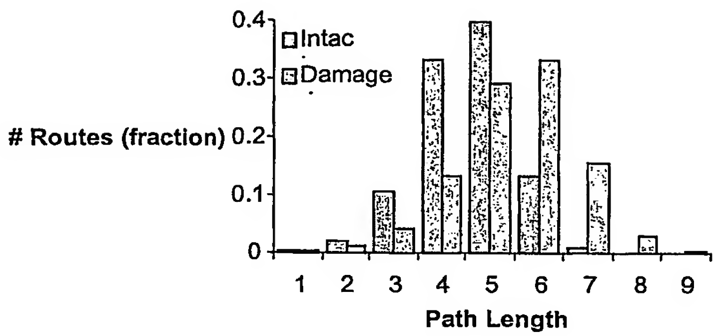


Figure 3a

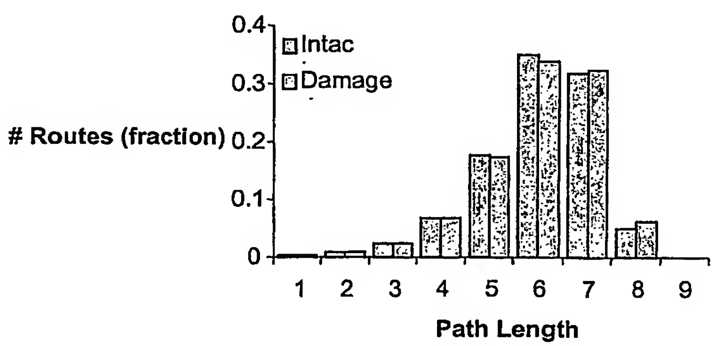


Figure 3b

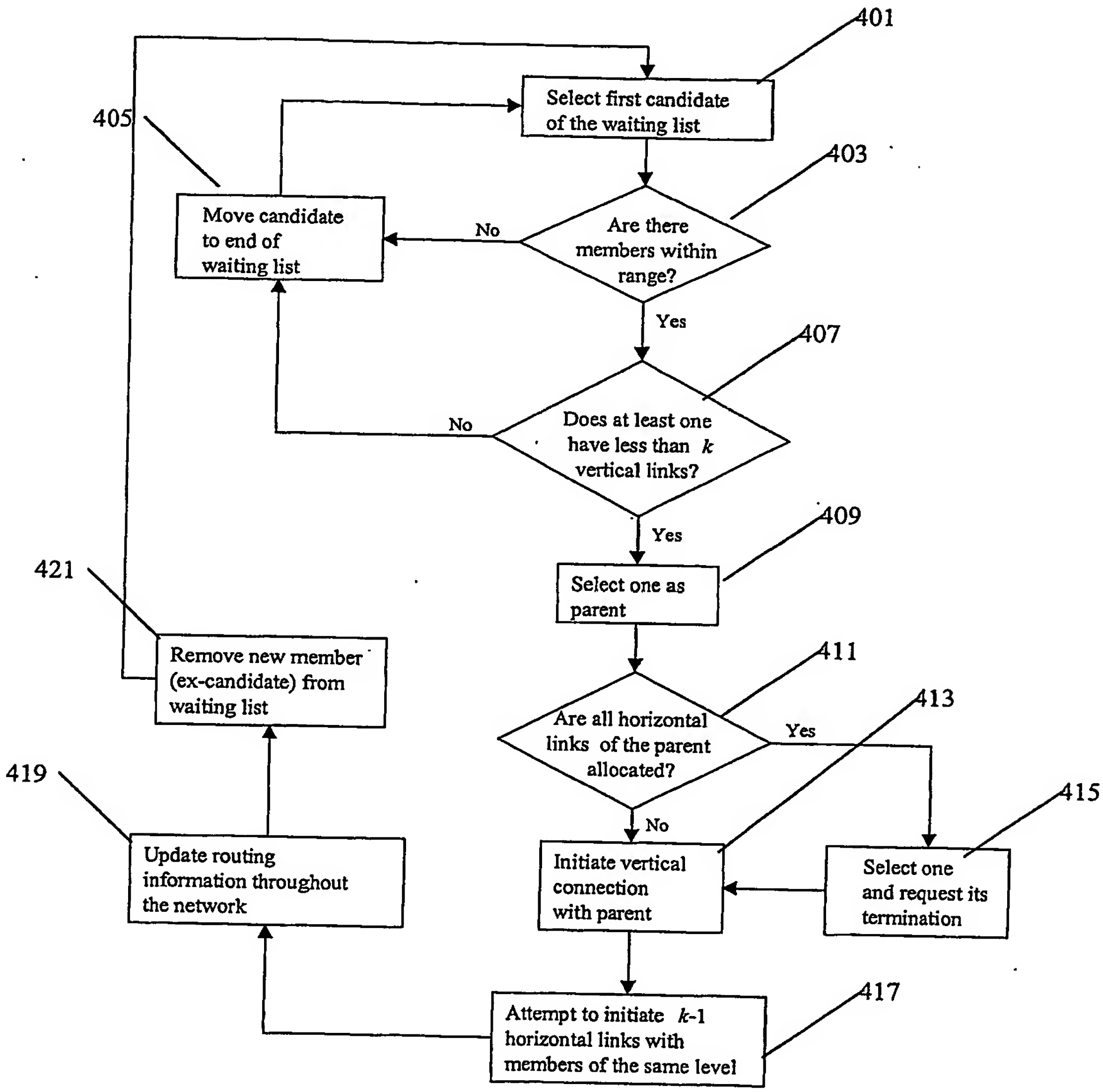

401
Select first candidate of the waiting list
403
Are there members within range?
No
405
Move candidate to end of waiting list
Yes
407
Does at least one have less than k vertical links?
No
Yes
409
Select one as parent
421
Remove new member (ex-candidate) from waiting list
411
Are all horizontal links of the parent allocated?
Yes
413
415
419
Update routing information throughout the network
No
Initiate vertical connection with parent
Select one and request its termination
417
Attempt to initiate k-1 horizontal links with members of the same level

Figure 4

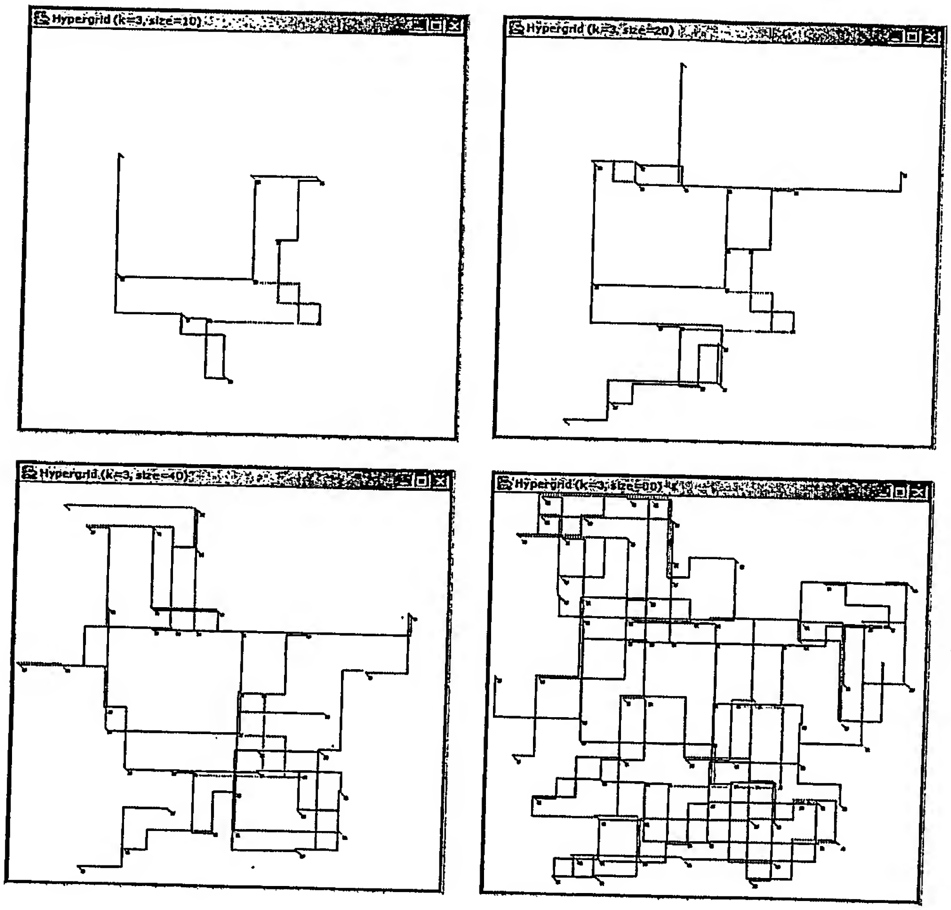
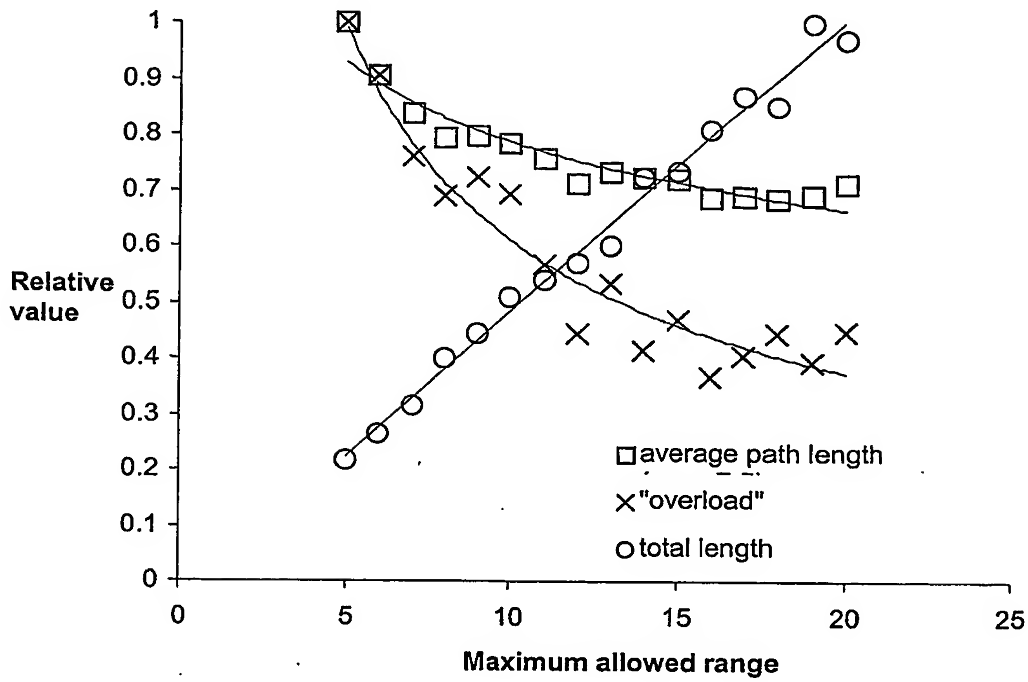

Hypergrid (k=3, size=10)
Hypergrid (k=3, size=20)
Hypergrid (k=3, size=40)

Figure 5



1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
Relative value
0
5
10
15
20
25
Maximum allowed range
average path length
"overload"
total length

Figure 6

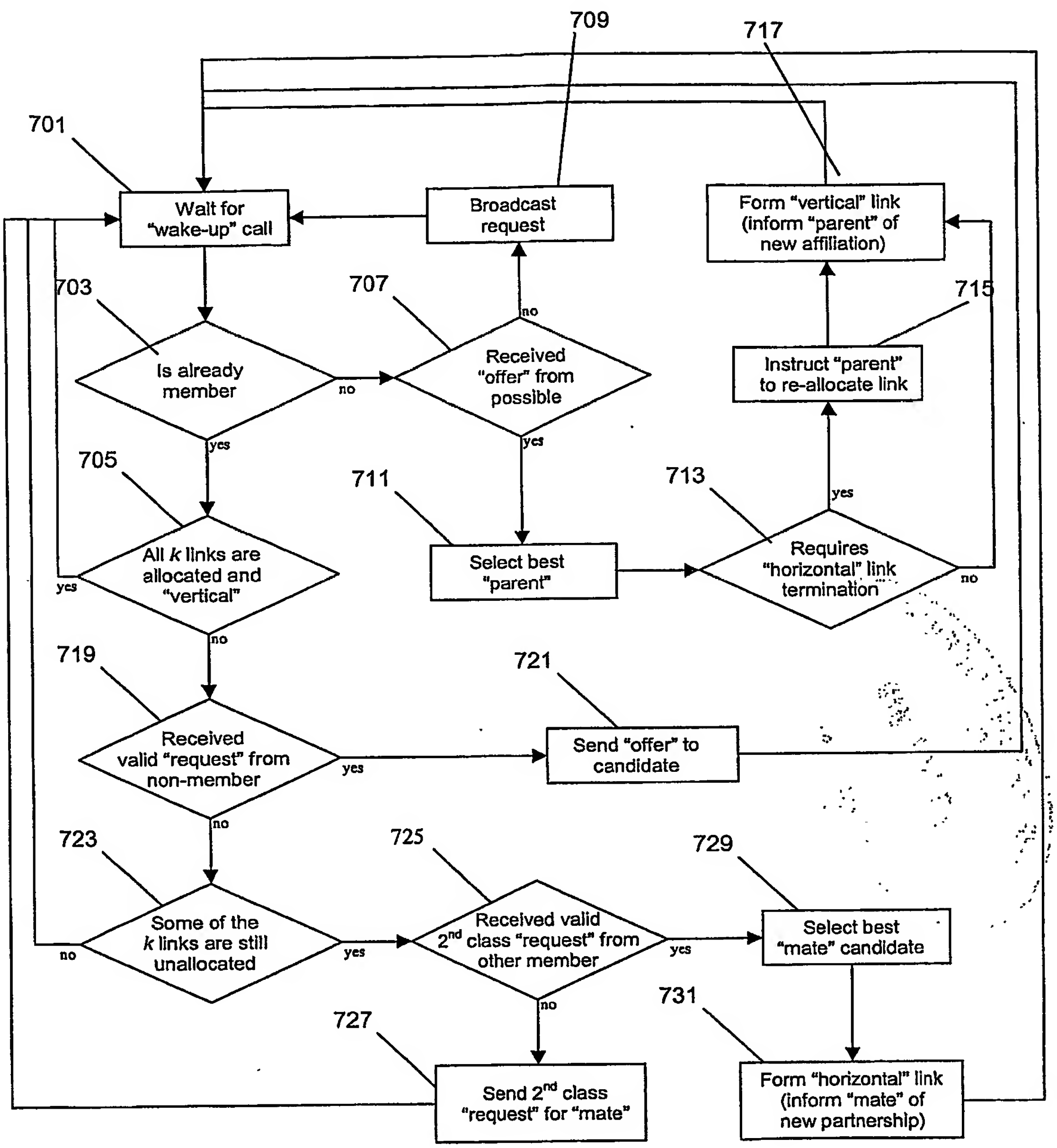

709
717
701
Wait for "wake-up" call
Broadcast request
Form "vertical" link (inform "parent" of new affiliation)
703
707
715
no
Is already member
no
Received "offer" from possible
Instruct "parent" to re-allocate link
yes
yes
705
711
713
yes
All k links are allocated and "vertical"
yes
Select best "parent"
Requires "horizontal" link termination
no
no
719
721
Received valid "request" from non-member
yes
Send "offer" to candidate
no
723
725
729
Some of the k links are still unallocated
no
yes
Received valid 2nd class "request" from other member
yes
Select best "mate" candidate
727
no
731
Send 2nd class "request" for "mate"
Form "horizontal" link (inform "mate" of new partnership)

Figure 7

Method and Apparatus for Network Management

The present invention relates to networks, in particular but not exclusively to computer or communications networks. The invention is particularly applicable in the organisation of network topology (connections).

It is known to share computer and other network resources (disk space, CPU time etc.) over a network. This arrangement enables a large group of simple devices with limited individual capabilities to provide an alternative to dedicated computers. This arrangement is often termed "grid computing" and enables the harnessing of the power of numerous networked machines scattered over distant geographical locations so as to be able to provide services on demand. These services may be provided using resources that would otherwise be under utilised. These grid computing arrangements can provide massive computing power at relatively low cost.

Other applications of distributed computing involve the connection of large numbers of low costs (perhaps recycled) PCs on a single physical location to provide an efficient (if large) supercomputer. However, as with all applications of distributed computing techniques, they can only be successful if the speed of data transmission matches that of data processing. In other words, it makes no sense to decompose the entire process of solving a complex problem in many simpler tasks if it is not possible to deliver intermediate results at the right place and time for the next step to proceed. Similarly, even a very fast search in a huge distributed database is useless if the retrieved information encounters a bottleneck on its way back to the source of the query.

Distributed computing systems are likely to operate best if not built according to a predefined plan. Such systems work best when they are allowed to grow and they do so in a generally unpredictable fashion. Similarly, the supercomputers built out of low-end and/or recycled components need to be capable of using any piece of hardware that becomes available. In both cases, the resulting network topology will be highly dynamic, where explicitly maintaining order (or even being able to discriminate between essential and non-essential components) will become impractical.

Current systems for sharing resources on a large scale such as in distributed computing systems that use non-specialised devices do not perform well when components of the system are removed, migrated or new components added. Often such activity requires a degree of redesign of the system architecture. Another problem with existing systems is that information flow can often become concentrated on components that are not well equipped to deal with such traffic thereby causing overloading.

A known way of supporting network growth is to upgrade components when the increasing workload exceeds their capacity. This is only practical as far as bottlenecks can be clearly identified, meaning they have to be stable in space and time (recurrent problems at a precise location, e.g. the hub of a particularly busy cluster in a hierarchical structure). In a fully decentralised system, traffic becomes so diffuse that it is difficult to isolate points of maximum stress, and/or so dynamic that such points are not associated with any specific network element. In these circumstances, ad-hoc replacement policies are seldom successful.

According to embodiments of the invention there is provided a novel network topology having connection rules allowing the network to grow to a desired size while respecting a set of constraints. The resulting network structure is one in which node degree is constant (all nodes have the same number of 1st neighbours) and the workload on the most busy member(s) (in terms of traffic) typically grows as a logarithmic function of network size. This is achieved by cross-allocating unused links within each level of the tree, until they are needed to provide an access point for newcomers. The cross allocated links may serve as shortcuts between (topologically) distant parts of the network, reducing its diameter and average path length, while re-routing some of the traffic away from the more busy (central) nodes.

Embodiments of the invention facilitate the addition, removal and migration of network components without the need for redesigning the entire architecture. This improves the robustness and plasticity of the network. Furthermore, information flow within the

network is as homogeneously distributed as information processing so as to generally avoid a situation where a small sub-set of network elements become primary relays. This makes the network more scalable.

Embodiments of the invention will now be described with reference to the accompanying drawings in which:

Figures 1a and 1b are schematic representations of a known network topology (tree) and a network according to the present invention respectively;

Figure 2 is a graph illustrating the traffic flows within the networks of figures 1a and 1b;

Figures 3a and 3b are graphs showing the performance of the networks of figures 1a and 1b in response to directed attack;

Figure 4 is a flow chart illustrating the process carried out during the process of joining a network in accordance with an embodiment of the invention;

Figure 5 is a schematic representation of a network being built using the process of figure 4;

Figure 6 is a graph illustrating the performance of the network built in accordance with the process of figure 4; and

Figure 7 is a flow chart illustrating the process carried out during the process of nodes joining a network in accordance with another embodiment of the invention.

Figure 1 is a schematic representation of a prior art network 101 of computers A to Q. The computers A to Q are capable of maintaining the same number (four) of connections as others. This connection limit prevents any one of the computers A to Q acting as a possible hub in the network. In this type of design, comprising no dedicated routers or relays, connecting from one computer to another over the network 101 involves making a series of connections between similar devices. In the network 101, there is only one route between any two of the computers A to Q. Also, node usage obeys a predictable pattern as long as traffic is homogeneously distributed between all computers A to Q. The closer one comes to the centre of the network i.e. computer A, the higher the information flow along the network links.

This traffic pattern means that computer or node A may have to handle 13 times more traffic than its least busy counterparts computers F, to Q. Assuming that all devices A to Q have similar capabilities, the "tree-like" design of network 101 appears susceptible to become overloaded. This demonstrates that imposing an upper limit on node connection (four in this example) does not reduce the chances of network overload. In fact, it appears that the opposite is the case. Adding this one local constraint (originally intended to lower pressure on supposedly limited devices) results in node A being forced to act as a hub in the network 101.

Detecting that a given node is likely to become a bottleneck may not always be feasible since it is not apparent from the number of connections that a node has. The overload of node A is relatively easy to observe when looking down at the schematic representation of the network 101 in figure 1a. However, from the viewpoint of individual nodes in the network or where no network representation exists, detecting potentially overloaded nodes or bottlenecks is more difficult. For example, in the network 101 nodes A to E all have the same number of first neighbours, so it is not obvious that node A will be liable to be overloaded.

The problem illustrated above with reference to figure 1a could easily occur in a network undergoing a decentralised growth process, whereby nodes with available connections advertise for other nodes to join the network. Early members of the network are likely to end up in the position of acting as core relays with newly comers gradually filling up empty spaces on the periphery of the network.

Figure 1b is a schematic representation of a network 103 in accordance with an embodiment of the present invention. The network 103 comprises interconnected nodes A to Q which is similar to the network of figure 1a. However, in the network 103 the connection rules for each node have been modified. In addition to each node being constrained by having a maximum number of connections, the peripheral nodes are not allowed to have fewer connections than the more central nodes. This results in the architecture shown in figure 1b. The design rules used to produce it specify that nodes should first be arranged in a tree. Then the remaining node connections are cross-allocated at random between peripheral nodes. The result is a network topology with a

typically very low clustering coefficient. In other words, the neighbours of a given nodes neighbouring nodes are not neighbours of the given node.

The resulting network topology in figure 1b has less traffic passing through the core than that of figure 1a. In the network 103, node A is part of only twice as many routes as any peripheral node: on average, nodes F to Q are part of approximately 26 such routes, compared to 50 for the "hub" node A. However, in network 101, 208 of the same 17x16 = 272 directed routes pass through node A.

The relatively homogeneous distribution of the workload shown for the topology of figure 1b is maintained in larger systems. Figure 2 is a graph showing the percentage of traffic through the central hub of a network against the size of that network. The graph shows the results of simulations of the network topologies described above with reference to figures 1a and 1b but on a larger scale. The graph also shows the results for a standard tree topology (figure 1a) by way of comparison. In the first simulation, the operation of a packet-switching network was modelled in which every node is sending 100 packets to randomly selected destinations, resulting in the total amount of information exchanged being a linear function of system size. The simulation demonstrated that in a topology of degree 4 (four connections per node) as in figure 1b, comprising 1457 nodes (7 layers), less than 1% of all packets sent along shortest routes still transit through the core. More precisely, the first simulation shows the workload on the hub to be a logarithmic function of the total number of nodes when the topology described with reference to figure 1b is adopted.

The second simulation was also carried out using the scale free "counterpart" of the network of figure 1b. This scale free topology is obtained by applying the preferred attachment rule for node connections, whereby the probability for a node to be selected as a "host" by a newly joining node is a linear function of the nodes degree. This results in some nodes having many more connections than others do. It is therefore a necessary feature of any scale-free network that node degree is not arbitrarily fixed. In other words, by "counterpart" means that the scale free network shares other key attributes of the network of the first simulation, namely same number of nodes (1457) and comparable number of connections (3000). The diameter of the

network is very similar in both cases (8 for the scale-free network versus 9 for the first simulation) even though the average path length is significantly different (4.53 versus 5.99 respectively). Yet in the scale-free simulation, close to 20% of the traffic is routed through the most highly connected node (the closest equivalent to the "hub"). Furthermore, comparison with smaller networks of similar design suggests that the workload on the main relay is nearly a linear function of the total number of nodes (it is actually a power law with the exponent slightly lower than 1, see figure 2).

It should be noted that each node in the network stores a variable called "height" which is used to indicate the position of the node in the network hierarchy. When a node joins the network, it sets its own "height" in the tree to that of its new parent plus one. As a result, as soon as a node joins the network it has a well-defined height in the hierarchy (the root or first node's height = 0, root's children's height = 1, root's children's children's height = 2 etc.). Links between nodes having the same height in the network are termed horizontal links, while links involving a hierarchical relationship are termed vertical links i.e. a parent-child link.

Comparing the performance of the topologies from the first and second simulations above shows that the topology of the second simulation, although marginally increasing the average number of hops between 2 randomly selected vertices, results in a large improvement in scalability. The central nodes would not have to support rapidly increasing traffic as the network grows, which is a major problem for large-scale distributed computing. Also, because in the topology of the second simulation, the constraints are exactly the same for any node that joins and at any time in the network's history, the connection rules are simple and easy to apply. These rules can be summarised as follows: in order to join a new node to the network of degree k (i.e. where each node has k connections) then the following steps should be carried out:

1. Identify the node with the lowest height (i.e. the innermost node) in the network that is maintaining horizontal connections.
2. Request one of these connections to be terminated and reallocated to the joining node, the link becoming vertical in the process.

3. Attempt to initiate k-1 horizontal links between the joining node and other nodes in the network having the same height as the joining node and which are advertising a spare connection.

Once this process is complete, the new node is a member of the network and if the network keeps growing, other layers will gradually form on top of the newly joined node but without adding significantly to the workload of the new node.

In order to compensate for the small increase in traffic that can occur when a node becomes increasingly submerged in the network, then in some embodiments a reward scheme may be implemented. In the scheme, submerged nodes obtain services at an incremental discount dependent on how far the surface of the network has moved away. Indeed, as the network's size grows faster than the workload on nodes, and considering the fact that the very principle of distributed computing is about sharing resources, it may become highly beneficial for a node to be more deeply submerged in the network. This would facilitate the replacement of departing nodes by their former subordinate nodes and initiate a cascade of inward migrations to restore the network's integrity.

Another important feature of network topology design is the resistance of the network to directed attack. The network topologies described above in relation to figures 1a and 1b have been subjected to simulations of directed attack by the periodic removal of nodes and the effect that this had on the possible routes through the network noted. Figure 3a shows the results for the directed attack simulation for the scale free network topology. As can bee seen from the graph, removing the 1% busiest nodes from the intact network has a considerable effect on path length distribution. Figure 3b shows the results of the directed attach on the network topology as outlined above in relation to figure 1b. In this case, the change in path length distribution is negligible. Furthermore, the redirected traffic is homogeneously distributed, resulting in the workload on surviving nodes being virtually unchanged (average ratio after/before attack is ~1.02, with a maximum of ~1.41) unlike in the scale free network (average ratio ~1.55, maximum ~6.84).

Figure 4 is a flow chart illustrating the algorithm for connecting nodes to build a network in accordance with the rules outlined above with reference to the second simulation. Figure 5 is a sequence of schematic representations of a physical network building itself in accordance with the algorithm of figure 4. The network is built on a lattice space of 20x20 with one cell out of four points (random distribution) containing a candidate member (i.e. density = 0.25). The network initiator is randomly chosen among all the candidate nodes and the entire structure is grown progressively in accordance with the algorithm of figure 4.

With reference to figure 4, at step 401, the network management system that initiates the network connection broadcasts a message asking for nodes that have spare links and builds a candidate list from the received replies. At step 403 a candidate is selected from the list and the system checks that the candidate is within range of a node that is a member of the network and if not then processing moves to step 405 at which the candidate is returned to the end of the list and another candidate selected at step 401.

If at step 403 the candidate is within range of a member node then processing moves to step 407 at which a check is carried out to establish whether the member node has less than k vertical links (where k is the degree of the network i.e. the maximum allowed number of links per node). If not the processing moves to step 405 and processing continues as described above from that step. If a free vertical link is identified in the member node then processing moves to step 409 where the member node is selected by the candidate node as its parent node. Also, at step 409 the candidate node sets its height to that of the parent plus one and processing moves to step 411.

At step 411, the parent links are inspected to establish whether all of its horizontal links are allocated. If all the horizontal links are allocated then processing moves to step 415 where the parent is requested to terminate one of those links and processing moves to step 413. If at step 411 unallocated horizontal links are identified then processing moves straight to step 413 at which a vertical link is initiated between the candidate node and the parent node. Processing then moves to step 417.

At step 417, the candidate node (now joined to the network) broadcasts a request for other nodes of the same height in the network with spare links to identify themselves. Those other nodes that respond are placed in a waiting list. The newly joined node then chooses one of the candidates from the list and forms a horizontal link with it. This process is repeated until the newly joined node has no spare links remaining and processing moves to step 419 at which the routing information held in the network is updated to take account of the new member and of the newly formed connections between the nodes. Processing then moves to step 421 where the newly joined node is removed from the waiting list and processing returns to step 401.

As noted above, figure 5 shows an example of a physical network (the term "physical" is used to mean that the location of the nodes on the blueprint in the figure is meant to represent the position of the nodes in real space, not their topological situation). The complexity of the architecture comes from the fact that nodes join in a random order and the entire network is grown while respecting the local constraints mentioned earlier. However from a topological point of view, the apparently highly disorganised network has the same underlying structure as the apparently tidier structure shown on Fig. 1b.

Figure 6 shows a graph illustrating the performance of the network described above with reference to figures 4 and 5. Assuming that horizontal links, when re-allocated, can be recycled only if they are long enough to reach their new endpoint, the cumulative length of the network is a linear function of the maximum range allowed between 1st neighbours. The average path length is inversely correlated with the same parameter. The graph also shows the variation of a global variable called "overload". It is based on the assumption that all nodes have identical capabilities and that the traffic should therefore ideally be evenly distributed between them. A network comprising N nodes obviously has $N^2/2$ shortest routes linking all of its members (provided self-targeting is allowed). Each node should therefore ideally not be part of more than $N/2$ such routes. The "overload" is the proportion of shortest routes that require some of the nodes they are made of to exceed this limit. Exceeding the limit is a cause for node stress and could result in bottlenecks forming in the network, so this complex variable should be kept as low as possible. The fact that it is inversely proportional to

maximum allowed range as well suggests that several factors must be considered when looking for a suitable compromise between minimising cost and maximising efficiency in a physical network.

The algorithm described above with reference to figure 4 shows the operation of a centralised system which ensures that new nodes join sequentially if the constraints on the maximum allowed range and link availability are satisfied. If a node is scheduled to join but the right conditions are not met (e.g. the distance to the nearest member is higher than the maximum authorised range), it is transferred to the waiting list. Another as yet unconnected candidate could provide a suitable entry point at a later stage of network development. However, all the connections are made under the control of the centralised network management system. Figure 7 represent an equivalent algorithm for carrying out essentially the same process in a fully decentralised system. In this arrangement member nodes and candidate nodes negotiate connections independently by exchanging a series of "request" and "offer" messages between each other. In other words there are no centralised decisions.

With reference to figure 7, each node sits idle (from the point of view of the connection process) at step 701 until a relevant message is received that activates the process. The node may also be arranged to activate itself at predetermined intervals to carry out a status check or other automated process. When a message is received, processing moves to step 703 at which the node establishes whether or not it is a member of the network and if so processing moves to step 705. At step 705, the node determines whether all of its links are allocated and are vertical. If this is the case then processing returns to step 701 and the node becomes idle again.

If at step 703 the node determines that it is not a member of the network processing moves to step 707 where it checks whether or not it has received an offer for connection to the network from a prospective parent node. If no such offer has been received then processing moves to step 709 where the node broadcasts a request to join the network and then becomes idle again at step 701 to await any replies. Any such reply would bring the process from step 701 to step 707 at which processing would then move on to step 711. At step 711 the node chooses one of the offers

received to join the network by linking to a parent node and processing moves to step 713.

At step 713 the node determines whether the parent needs to terminate on of its horizontal links in order to provide a connecting point for the node and if this is the case processing moves to step 715 where the request to terminate a link is made to the parent. The parent node also initiates a process with the node to which the terminated link was connected to inform that other node of that termination. If at step 713 a free link is identified then processing moves straight to step 717. At step 717 the connection is made between the joining node and the parent and the newly joined node sets its height to that of the parent plus one. Processing then returns to step 701.

If at step 705 the node determines that is has a free link then processing moves to step 719 where it checks to see if a request to join the network has been received from a non member. If this is the case then processing moves to step 721 where an offer for connection is sent to the requesting node and processing returns to step 701 to await any response. If at step 719 no requests have been received then processing moves to step 723 where the node check whether or node any of its links are unallocated and of not processing returns to step 701. If however links do remain unallocated then processing moves to step 725.

At step 725 the node checks to see if it has received any requests for connection from other members of the network (to form a horizontal connection). Such requests are treated with a lower priority (second class) than requests from non members i.e. a request for a parent node (first class requests). If no such low priority requests have been received then processing moves to step 727 where the node broadcasts a connection request to the other nodes in the network (a second class request) and processing returns to step 701 to await any reply. If at step 725 low priority requests have been received then processing moves to step 729 where one of the requests is selected. Processing then moves to step 731 where a horizontal link is initiated with the other node (mate) and processing returns to step 701 to the idle state.

The system described above for connecting nodes in a network can also be used as a connection protocol for generating a virtual network independently of the supporting

media and of the actual topology of the physical layer (i.e. organise hyperlinks). The system can also be used to create and manage a physical network such as a small to medium sized network (in terms of surface), perhaps featuring high component density and turnover. The system could be used in conjunction with adaptive topology to ensure that the cost of rewiring is maintained within acceptable limits (due to the limited spatial extension of the system). Possible examples of such networks could include highly dynamic local area networks where resources have to be shared but dedicated servers/routers are not considered an option or "junk" supercomputing facilities with high failure rate of component parts.

Both arrangements above can be implemented using network cards fitted with a number of sockets similar to the intended degree of the network. Cables can then simply be plugged and un-plugged as components are added to, transferred within or removed from the network. Adding a new piece of hardware is effected by locating an available entry point in the vicinity of the new device (unplugging and reallocating a "horizontal" cable if necessary) then plugging up to k-1 open-ended cables of the same topological layer into the new device's network card. Alternatively, programmable hardware can be used which would allow reconfiguring network topology without having to physically manipulate operational connections to restore system integrity.

It will be understood by those skilled in the art that the apparatus that embodies the invention could be a general purpose device having software arranged to provide an embodiment of the invention. The device could be a single device or a group of devices and the software could be a single program or a set of programs. Furthermore, any or all of the software used to implement the invention can be contained on various transmission and/or storage mediums such as a floppy disc, CD-ROM, or magnetic tape so that the program can be loaded onto one or more general purpose devices or could be downloaded over a network using a suitable transmission medium.

Unless the context clearly requires otherwise, throughout the description and the claims, the words "comprise", "comprising" and the like are to be construed in an inclusive as opposed to an exclusive or exhaustive sense; that is to say, in the sense of "including, but not limited to".

ABSTRACT

Method and Apparatus for Network Management

A method and apparatus for network management are disclosed in which nodes in the network are arranged to initiate links across the network tree structure. As a result, nodes are linked to their sibling nodes in addition to being lined to parent and child nodes.

Figure (1b)